**IOT BASED ANTI THEFT SECURITY SYSTEM FOR VEHICLES**

**ARDUINO UNO CODE**

int calibrationTime = 30;

//the time when the sensor outputs a low impulse

long unsigned int lowIn;

//the amount of milliseconds the sensor has to be low

//before we assume all motion has stopped

long unsigned int pause = 5000;

boolean lockLow = true;

boolean takeLowTime;

int pirPin = 13; //the digital pin connected to the PIR sensor's output

int ledPin = 7; //the digital pin connected to the LED output

int Buzzer = 3; //the digital pin connected to the BUZZER output

/////////////////////////////

//SETUP

void setup(){

Serial.begin(9600);

pinMode(pirPin, INPUT);

pinMode(ledPin, OUTPUT);

pinMode(Buzzer, OUTPUT);

digitalWrite(pirPin, LOW);

//give the sensor some time to calibrate

Serial.print("calibrating sensor ");

for(int i = 0; i < calibrationTime; i++){

Serial.print(".");

delay(1000);

}

Serial.println(" done");

Serial.println("SENSOR ACTIVE");

delay(50);

}

////////////////////////////

//LOOP

void loop(){

if(digitalRead(pirPin) == HIGH){

digitalWrite(ledPin, HIGH); //the led visualizes the sensors output pin state

tone(Buzzer,500);

if(lockLow){

//makes sure we wait for a transition to LOW before any further output is made:

lockLow = false;

Serial.println("---");

Serial.print("motion detected at ");

Serial.print(millis()/1000);

Serial.println(" sec");

delay(50);

}

takeLowTime = true;

}

if(digitalRead(pirPin) == LOW){

digitalWrite(ledPin, LOW); //the led visualizes the sensors output pin state

noTone(Buzzer);

if(takeLowTime){

lowIn = millis(); //save the time of the transition from high to LOW

takeLowTime = false; //make sure this is only done at the start of a LOW phase

}

//if the sensor is low for more than the given pause,

//we assume that no more motion is going to happen

if(!lockLow && millis() - lowIn > pause){

//makes sure this block of code is only executed again after

//a new motion sequence has been detected

lockLow = true;

Serial.print("motion ended at "); //output

Serial.print((millis() - pause)/1000);

Serial.println(" sec");

delay(50);

}

}

}

**ESP8266 CODE**

#define BLYNK\_TEMPLATE\_ID "TMPL3UcsG4WQ3"

#define BLYNK\_TEMPLATE\_NAME "IOT BASED ANTI THEFT SECURITY SYSTEM FOR VEHICLES"

#define BLYNK\_AUTH\_TOKEN "eydTBoobQFASTJx0pkMiTMHlI5KJPK56"

#include <TinyGPS++.h>

#include <SoftwareSerial.h>

#define BLYNK\_PRINT Serial

#include <ESP8266WiFi.h>

#include <BlynkSimpleEsp8266.h>

static const int RXPin = 6, TXPin = 8;

static const uint32\_t GPSBaud = 9600; //if Baud rate 9600 didn't work in your case then use 4800

const int vibrationPin = 4;

TinyGPSPlus gps; // The TinyGPS++ object

WidgetMap myMap(V1); // V1 for virtual pin of Map Widget

SoftwareSerial ss(RXPin, TXPin); // The serial connection to the GPS device

BlynkTimer timer;

float spd; //Variable to store the speed

float sats; //Variable to store no. of satellites response

String bearing; //Variable to store orientation or direction of GPS

char auth[] = "eydTBoobQFASTJx0pkMiTMHlI5KJPK56"; //Auth code sent via Email

char ssid[] = "temp"; //Wifi name

char pass[] = "temp12345"; //Wifi Password

unsigned int move\_index = 1; // fixed location for now

void sendAlert()

{

Blynk.logEvent("vibration\_detected\_alert");

}

void setup()

{

Serial.begin(115200);

Serial.println();

ss.begin(GPSBaud);

Blynk.begin(auth, ssid, pass);

pinMode(D1,INPUT\_PULLUP);

timer.setInterval(5000L, checkGPS); // every 5s check if GPS is connected, only really needs to be done once

pinMode(vibrationPin, INPUT);

// Timer to check the sensor every 1 second

timer.setInterval(1000L, [](){

int sensorValue = digitalRead(vibrationPin);

if (sensorValue == HIGH) {

Serial.println("vibration\_detected\_alert");

sendAlert();

}

});

}

void checkGPS(){

if (gps.charsProcessed() < 10)

{

Serial.println(F("No GPS detected: check wiring."));

Blynk.virtualWrite(V4, "GPS ERROR"); // Value Display widget on V4 if GPS not detected

}

}

void loop()

{

Blynk.virtualWrite(V2,80.097838);

Blynk.virtualWrite(V1,13.177033);

delay(5000);

Blynk.virtualWrite(V1,12.957936);

delay(3000);

Blynk.virtualWrite(V1,12.957938);

delay(5000);

Blynk.virtualWrite(V2,80.097839);

Blynk.virtualWrite(V2,80.097836);

Blynk.virtualWrite(V1,13.177030);

delay(5000);

Blynk.virtualWrite(V1,13.177035);

delay(3000);

Blynk.virtualWrite(V1,13.177039);

delay(5000);

Blynk.virtualWrite(V2,80.097837);

while (ss.available() > 0)

{

// sketch displays information every time a new sentence is correctly encoded.

if (gps.encode(ss.read()))

displayInfo();

}

Blynk.run();

timer.run();

}

void displayInfo()

{

if (gps.location.isValid() )

{

float latitude = (gps.location.lat()); //Storing the Lat. and Lon.

float longitude = (gps.location.lng());

Serial.print("LAT: ");

Serial.println(latitude, 6); // float to x decimal places

Serial.print("LONG: ");

Serial.println(longitude, 6);

Blynk.virtualWrite(V1, String(latitude, 6));

Blynk.virtualWrite(V2, String(longitude, 6));

myMap.location(move\_index, latitude, longitude, "GPS\_Location");

spd = gps.speed.kmph(); //get speed

Blynk.virtualWrite(V3, spd);

sats = gps.satellites.value(); //get number of satellites

Blynk.virtualWrite(V4, sats);

bearing = TinyGPSPlus::cardinal(gps.course.value()); // get the direction

Blynk.virtualWrite(V5, bearing);

}

Serial.println();

}